Programação Orientada a Objetos

Trabalho Prático

2021-2022

Sérgio Costa – 2020129026

Diogo Santos – 2020131432

Licenciatura de Engenharia Informática

Coimbra, 16 de janeiro de 2020

Índice

Introdução

Este trabalho consiste em conceber, um simulador/jogo de construção e desenvolvimento. O jogador terá a concessão de uma ilha e o jogador deve desenvolver essa ilha, industrializando-a e construindo todo um complexo fabril.

Criação e Inicialização das Classes

Criação da Ilha

Para a criação do "mapa da ilha", optamos pela criação de um array bidimensional de memoria dinâmica.

Criação de Zonas

Para a criação de zonas optamos por criar zonas dinâmicas ficando o array bidimensional assim:

*Zona ***arrayZonas;*

Ou seja as zonas são criadas dinamicamente e escolhidas aleatoriamente, confirmando que pelo menos existe uma zona de cada tipo.

Criação de Trabalhadores

Para a criação de trabalhadores optamos por criar um vector de trabalhadores e ficou com este aspeto:

*std::vector <Trabalhador *> trabalhadores;*

Ou seja os trabalhadores também são criados com memoria dinâmica. Optamos por criar este vetor na ilha, já que os trabalhadores podem ser movimentados de zona em zona.

Criação de Edifícios

Para a criação de edifícios optamos por criar apenas um edifício por zona e ficou com este aspeto:

Edificio edificio;*

Ou seja os edificios também são criados com memoria dinâmica. Optamos por criar este edifício dentro da zona na ilha, já que cada zona pode ter apenas um ou nenhum edifício e não pode ser movido entre zonas.

Criação de Recursos

Para a criação de recursos optamos por criar um vector de recursos por zona e ficou com este aspeto:

*std::vector <Recurso *> recursos;*

Ou seja os recursos também são criados com memoria dinâmica. Optamos por criar este vector dentro da zona na ilha, já que cada zona pode ter vários tipos de recursos.

Polimorfismo e Hierarquia

Edifícios

Nós optamos por usar hierarquia e polimorfismo nos edifícios. A partir da base, edifício, foram criadas as seguintes derivadas:

- Bateria
- Central elétrica
- Fundição
- Mina de Carvão
- Mina de Ferro
- Serração (ou edifício X)

As basses listadas acima deram origem a header files, com varias funções diferentes.

Zonas

Nós optamos por usar hierarquia e polimorfismo nas zonas. A partir da base, zona, foram criadas as seguintes derivadas:

- Deserto
- Floresta
- Montanha
- Pantano
- Pastagem
- Praia (ou zona X)

As basses listadas acima deram origem a header files, com várias funções diferentes. A nossa opção para a a zona X foi a criação de uma zona praia, que pelo sossego e paz, não permite que nenhum trabalhador peça demissão. São aceitos todos os edifícios mas devido as mares altas, os edifícios vão sendo destruídos e a partir do nono dia os edifícios desabam porque a areia esta muito molhada e os edifícios ficam fracos e desabam.

Estrutura do Projeto

O projeto para melhor organização esta dividido em pastas, por isso o projeto ficou com aa seguintes pastas.

Comandos

Para facilitar a comparação entre o comando escrito pelo utilizador e os comandos disponíveis optamos pela criação de um vetor de comandos com a estrutura seguinte *{<nome>, <descrição>, <argumento 1> <argumento 2> … <argumento N>}*
Para isso criamos uma classe chamada Comando com a estrutura de um comando, como esta indicado acima.

Edifício

Os edifícios como foi descrito acima foram criados usando polimorfismo e hierarquia. A base guarda o nome, estado (true/false) e o preço. Alem dos set e gets foram criadas as seguintes funções do tipo virtual, que são redefinidas nas derivadas caso necessário:

- *virtual float getArmazem();*
- *virtual float setArmazem(float n);*
- *virtual float setArmazemReduz(float n);*
- *virtual float setArmazemDobra(float n);*
- *virtual float upgradeNivel(float dinheiro);*
- *virtual float getProbDesabar();*

Ilha

Na ilha são declaradas grande parte das funções, como a contratação de trabalhadores, criação de zonas, comandos, funções de amanhecer e anoitecer, entre outras.

Interface

Na interface estão dois grandes ficheiros, a Main Window e a Game Window. Como este projeto usa o QT Widgets, os ficheiros .ui, são os editáveis para poder mudar o aspeto da interface gráfica. A Main Window vai pedir ao utilizador os parâmetros para a criação da ilha (linha e coluna) e fazer a respetiva validação, assim que os valores estejam corretos sai a Main Window e entra a Game Window. A Game window tem uma barra para introduzir os comando, o botão ou enter para executá-lo, abaixo esta a representação da ilha e do lado direito a informação de cada dia, que é atualizada quando é feio o comando *next*.

Recursos

Os recursos são usados nas zonas, nesta pasta esta guardada a classe que permite guardar o nome, preço e quantidade.

Trabalhadores

Os recursos são usados na ilha, nesta pasta esta guardada a classe que permite guardar o nome, id, posição, preço e probabilidade de se despedir.

Zona

Na zona tal como na ilha são declaradas grande parte das funções, como a manipulação de recursos, criação e manipulação de edifícios, entre outras.

QT

Introdução

O Qt é um framework para desenvolvimento de interfaces gráficas em C++. Com ele é possível desenvolver aplicações uma única vez e compilá-los para diversas plataformas sem que seja necessário alterar o código fonte. Neste projeto eu utilizei o QT Widgets no CLion.

Instalação

A versão que eu instalei foi a 5.12.12, e open source, no seguinte caminho do meu computador:

C:\Qt\5.12.12

Utilização no CLion

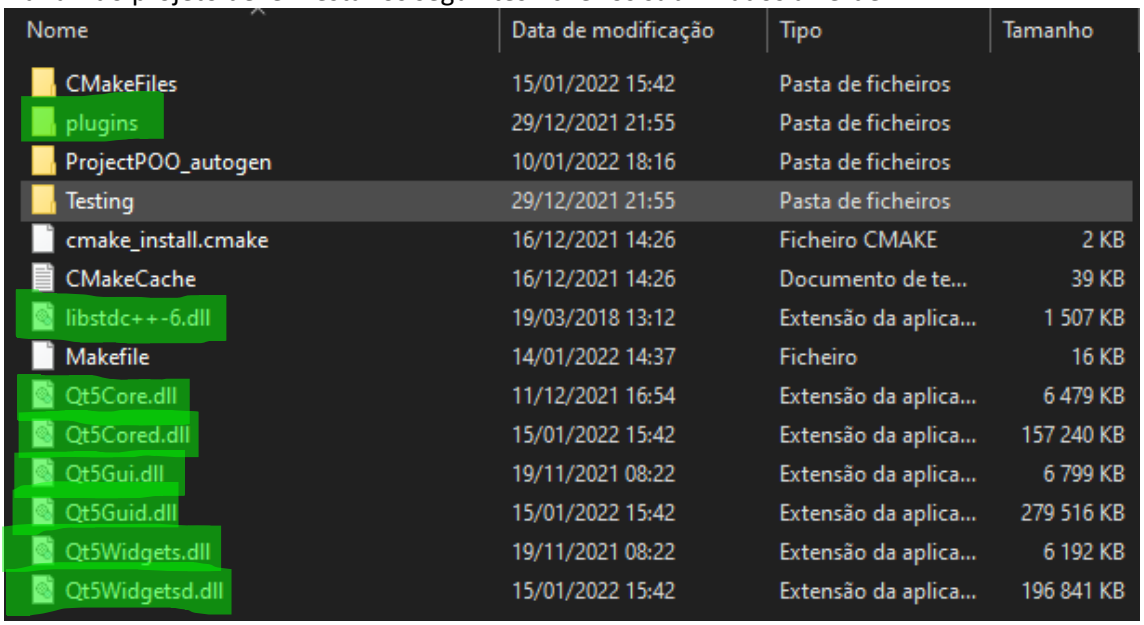
Nas definições → Build, execution, deppoyment → CMake e em CMake Options, acrescentar o caminho, no meu caso fica assim:

-DCMAKE_PREFIX_PATH=C:\Qt\5.12.12\mingw73_32\lib\cmake

Tambem no Run/Debug configurations foi precisso mudar a Environment variables para:

QT_QPA_PLATFORM_PLUGIN_PATH=C:\Qt\5.12.12\mingw73_32\plugins\platforms

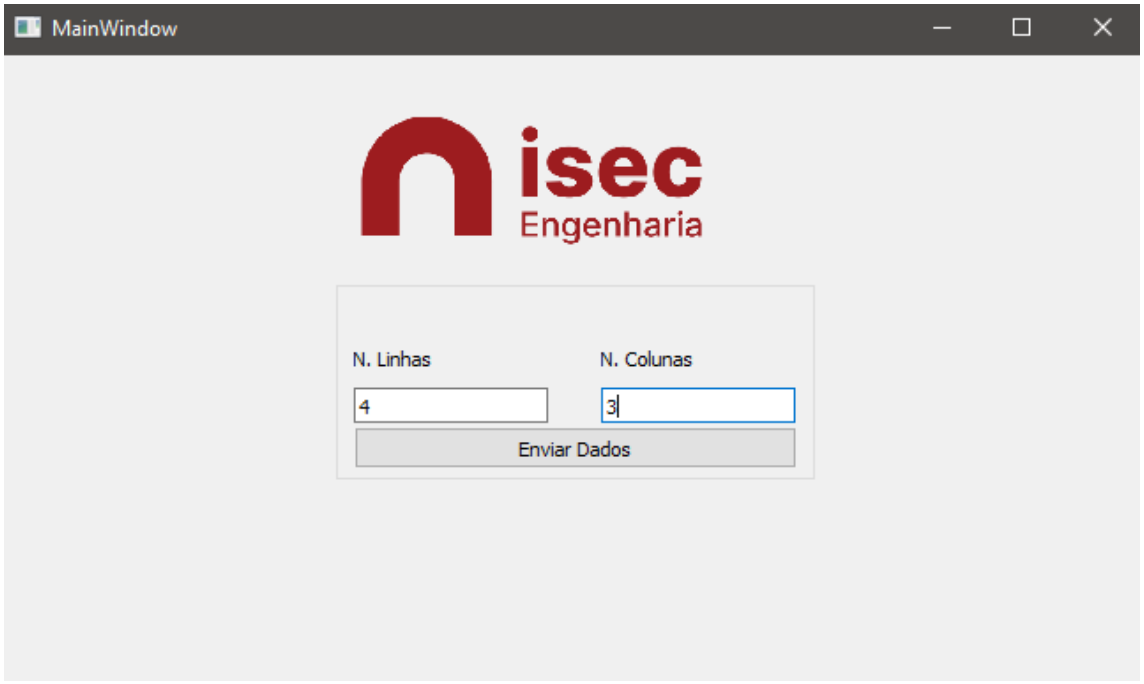
Na raiz do projeto devem estar os seguintes ficheiros sublinhados a verde:

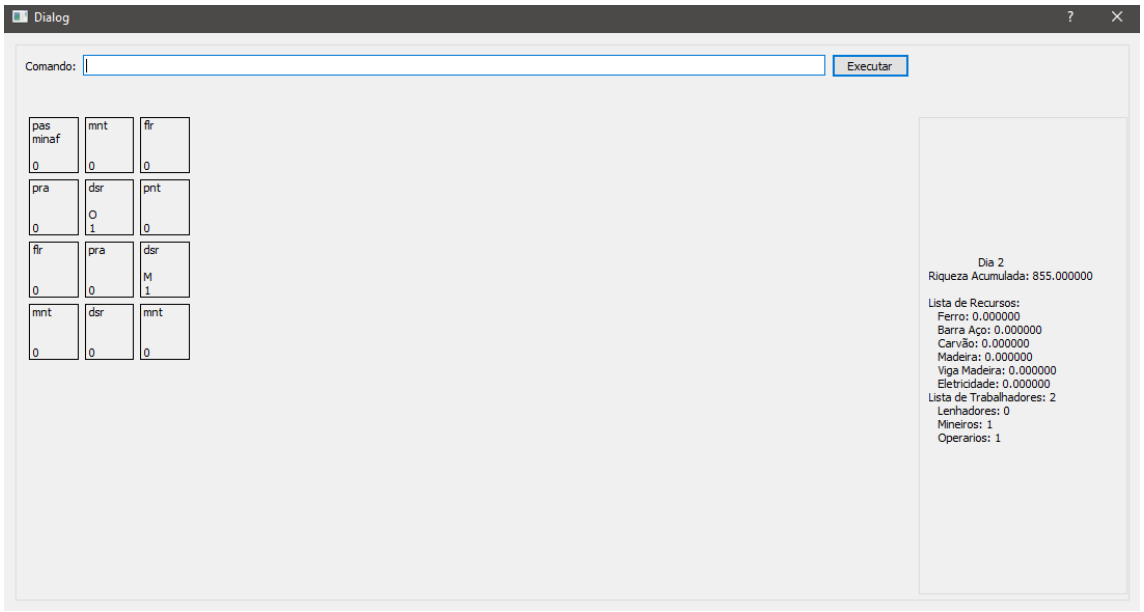


Nome	Data de modificação	Tipo	Tamanho
CMakeFiles	15/01/2022 15:42	Pasta de ficheiros	
plugins	29/12/2021 21:55	Pasta de ficheiros	
ProjectPOO_autogen	10/01/2022 18:16	Pasta de ficheiros	
Testing	29/12/2021 21:55	Pasta de ficheiros	
cmake_install.cmake	16/12/2021 14:26	Ficheiro CMAKE	2 KB
CMakeCache	16/12/2021 14:26	Documento de te...	39 KB
libstdc++-6.dll	19/03/2018 13:12	Extensão da aplica...	1 507 KB
Makefile	14/01/2022 14:37	Ficheiro	16 KB
Qt5Core.dll	11/12/2021 16:54	Extensão da aplica...	6 479 KB
Qt5Cored.dll	15/01/2022 15:42	Extensão da aplica...	157 240 KB
Qt5Gui.dll	19/11/2021 08:22	Extensão da aplica...	6 799 KB
Qt5Guid.dll	15/01/2022 15:42	Extensão da aplica...	279 516 KB
Qt5Widgets.dll	19/11/2021 08:22	Extensão da aplica...	6 192 KB
Qt5Widgetsd.dll	15/01/2022 15:42	Extensão da aplica...	196 841 KB

Dentro da pasta plugins tem de ter outra com o nome platforms e dentro dessa pasta tem de estar a dll, qwindowsd.dll (link para aceder aos ficheiros: aqui)

Aplicação


MainWindow
isec
Engenharia
N. Linhas
N. Colunas
4
3
Enviar Dados


Dialog
Comando:
Executar
pas minaf 0
mnt 0
flr 0
pra 0
dsr O 1
pnt 0
flr 0
pra 0
dsr M 1
mnt 0
dsr 0
mnt 0
Dia 2
Riqueza Acumulada: 855.000000
Lista de Recursos:
Ferro: 0.000000
Barra Aço: 0.000000
Carvão: 0.000000
Madeira: 0.000000
Viga Madeira: 0.000000
Eletricidade: 0.000000
Lista de Trabalhadores: 2
Lenhadores: 0
Mineiros: 1
Operarios: 1